FIG. 1

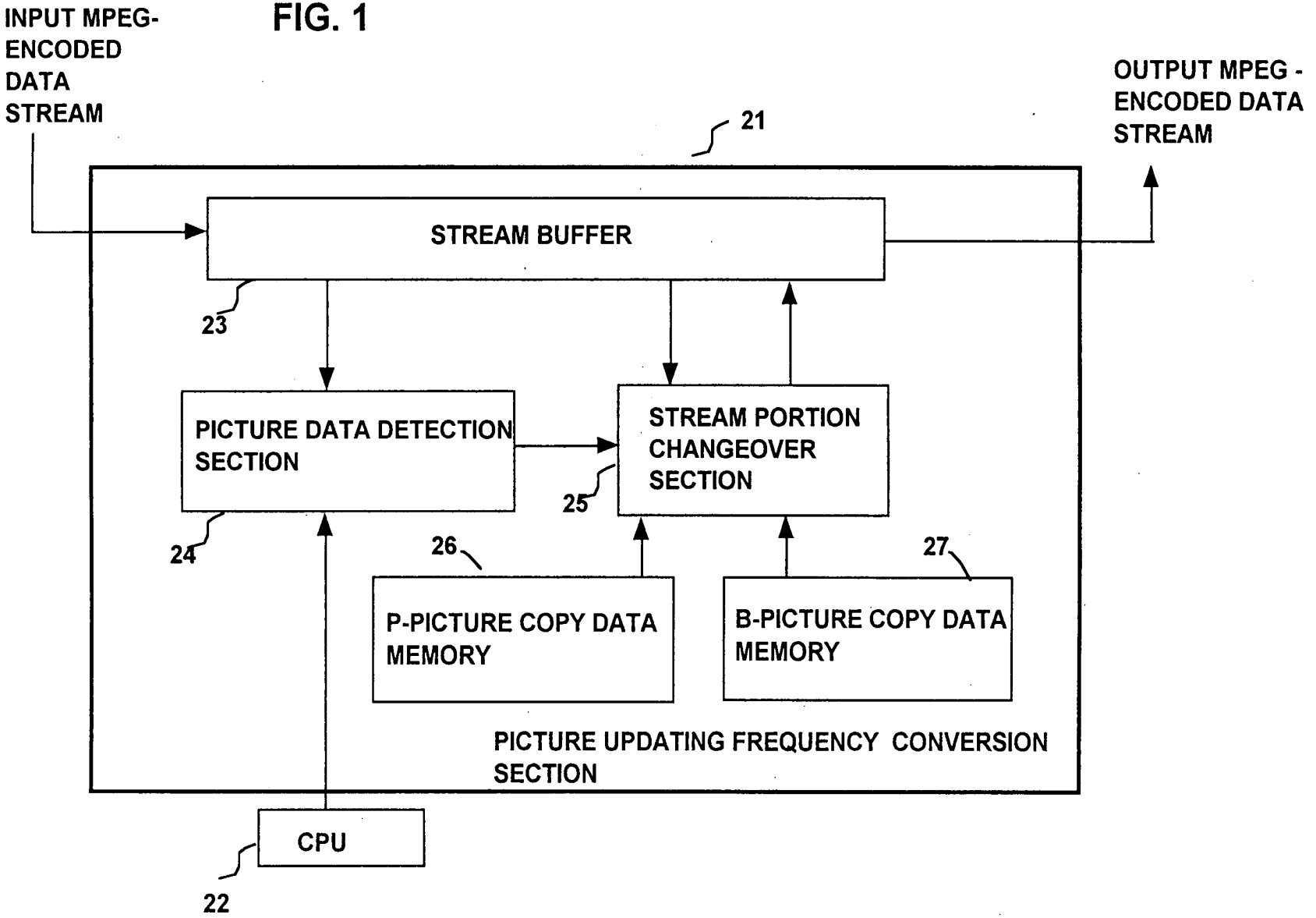
INPUT MPEG-ENCODED DATA STREAM
OUTPUT MPEG - ENCODED DATA STREAM
21
STREAM BUFFER
23
PICTURE DATA DETECTION SECTION
STREAM PORTION CHANGEOVER SECTION
25
24
26
27
P-PICTURE COPY DATA MEMORY
B-PICTURE COPY DATA MEMORY
PICTURE UPDATING FREQUENCY CONVERSION SECTION
CPU
22

FIG. 2

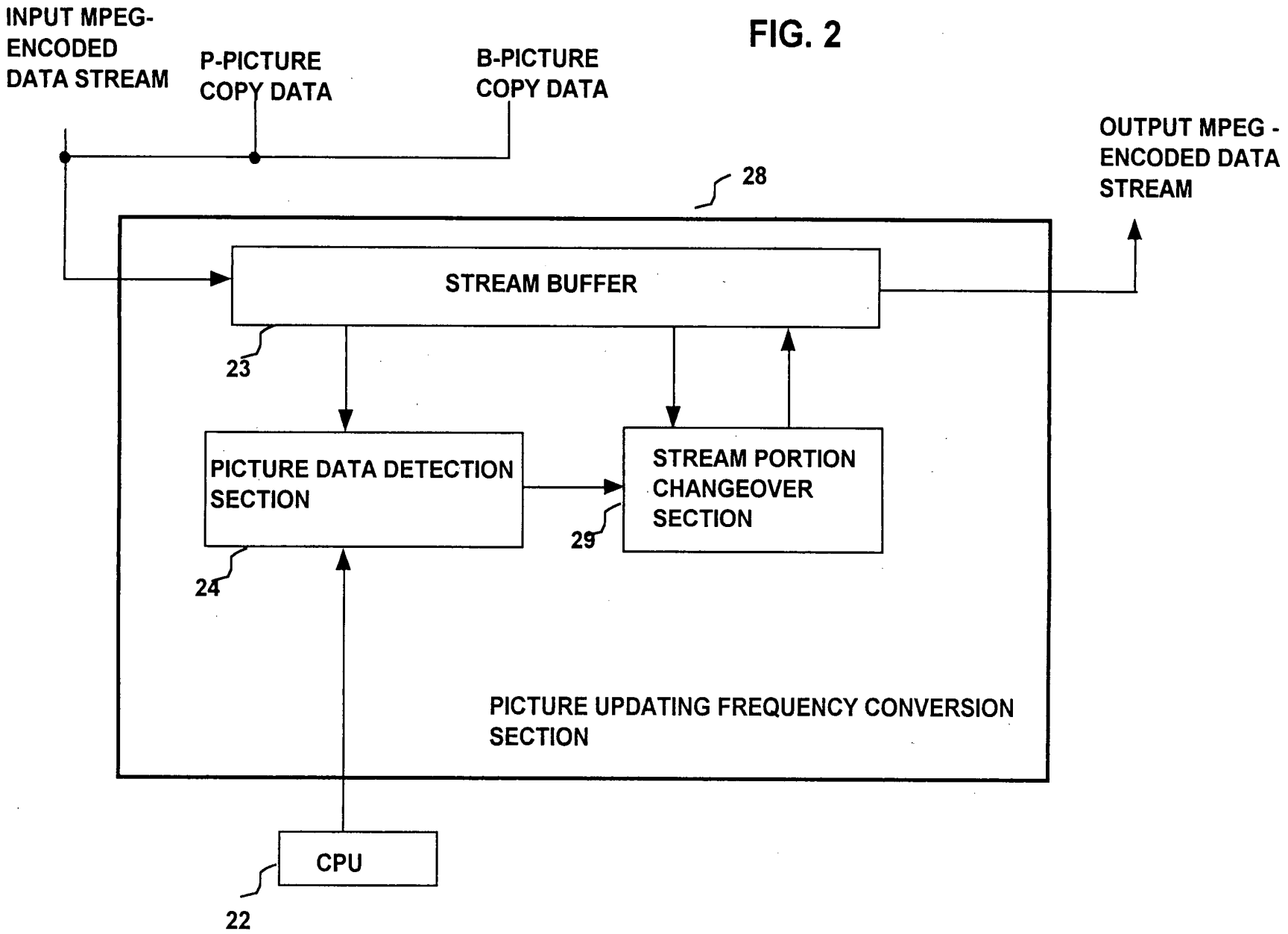
INPUT MPEG-ENCODED DATA STREAM
P-PICTURE COPY DATA
B-PICTURE COPY DATA
OUTPUT MPEG - ENCODED DATA STREAM
28
STREAM BUFFER
23
PICTURE DATA DETECTION SECTION
24
STREAM PORTION CHANGEOVER SECTION
29
PICTURE UPDATING FREQUENCY CONVERSION SECTION
CPU
22

FIG. 3

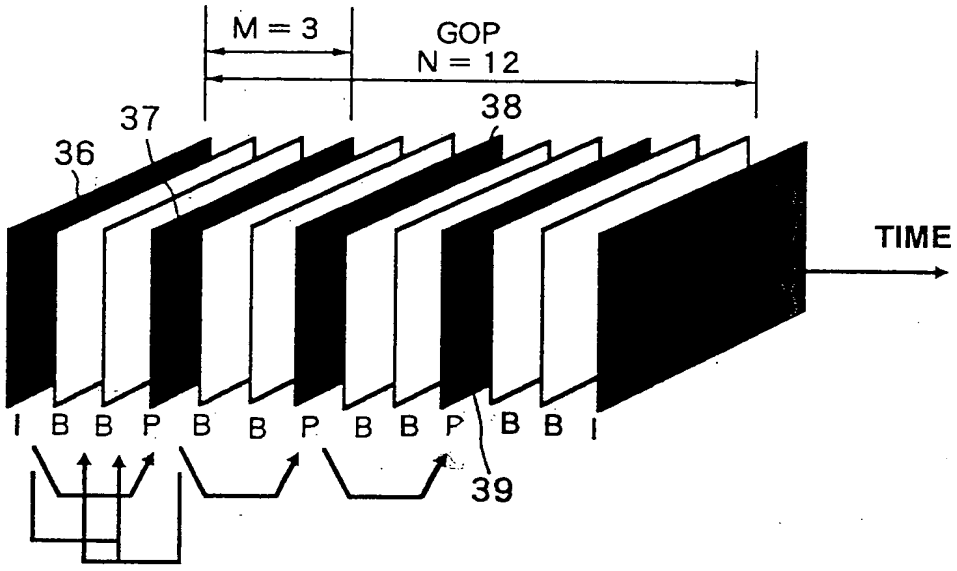
M = 3
GOP
N = 12
37
38
36
TIME
I B B P B B P B B P B B I
39

FIG. 4

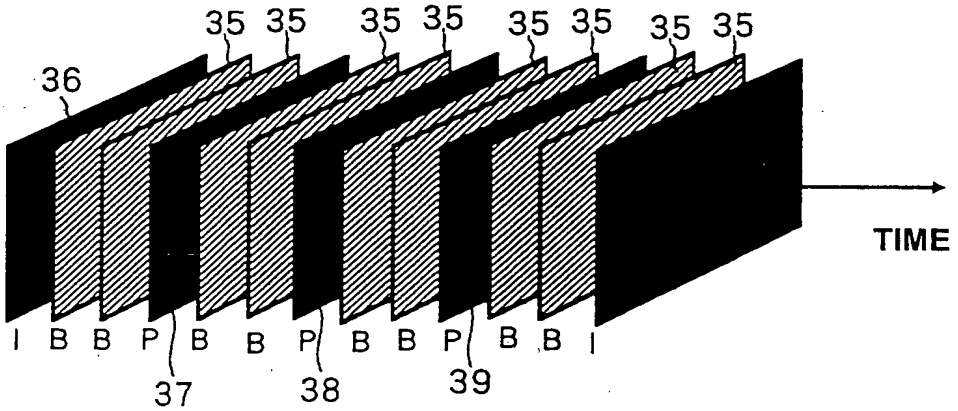
35 35 35 35 35 35 35 35
36
TIME
I B B P B B P B B P B B I
37
38
39

FIG. 5

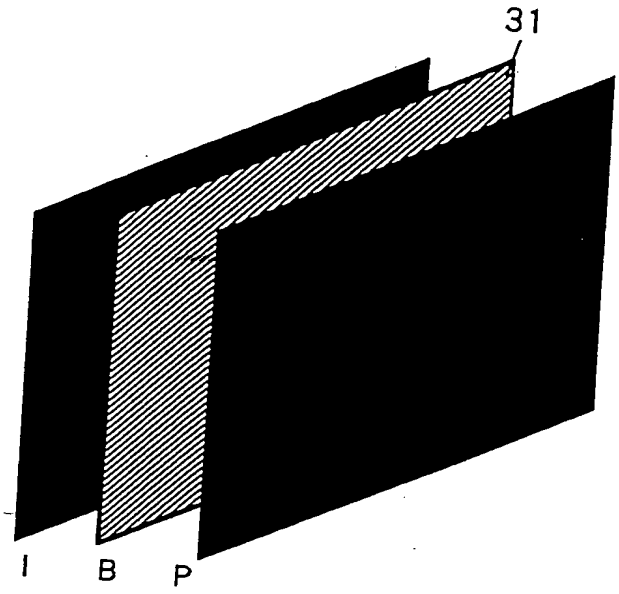

FIG. 6

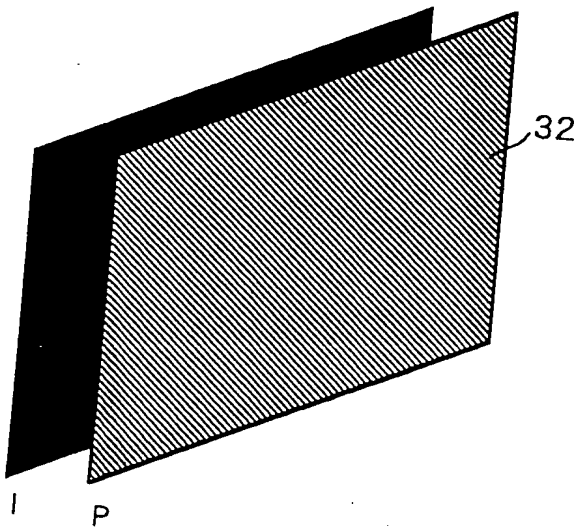

FIG. 7

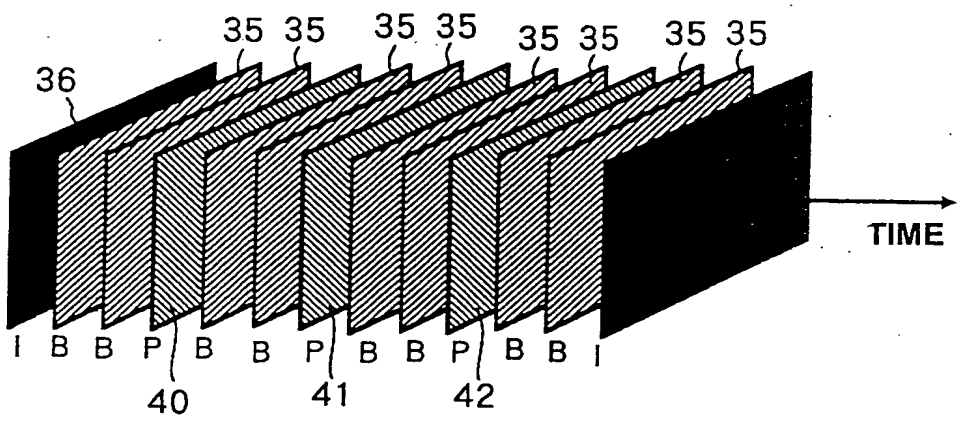
35 35 35 35 35 35 35 35
36
TIME
I B B P B B P B B P B B I
40
41
42

FIG. 8

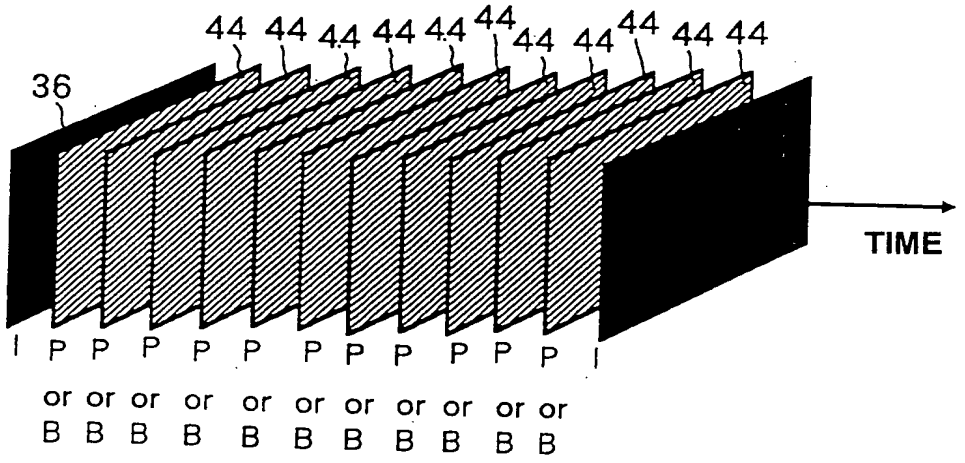
44 44 44 44 44 44 44 44 44 44 44
36
TIME
I P P P P P P P P P P I
or or or or or or or or or or or
B B B B B B B B B B B

FIG. 9

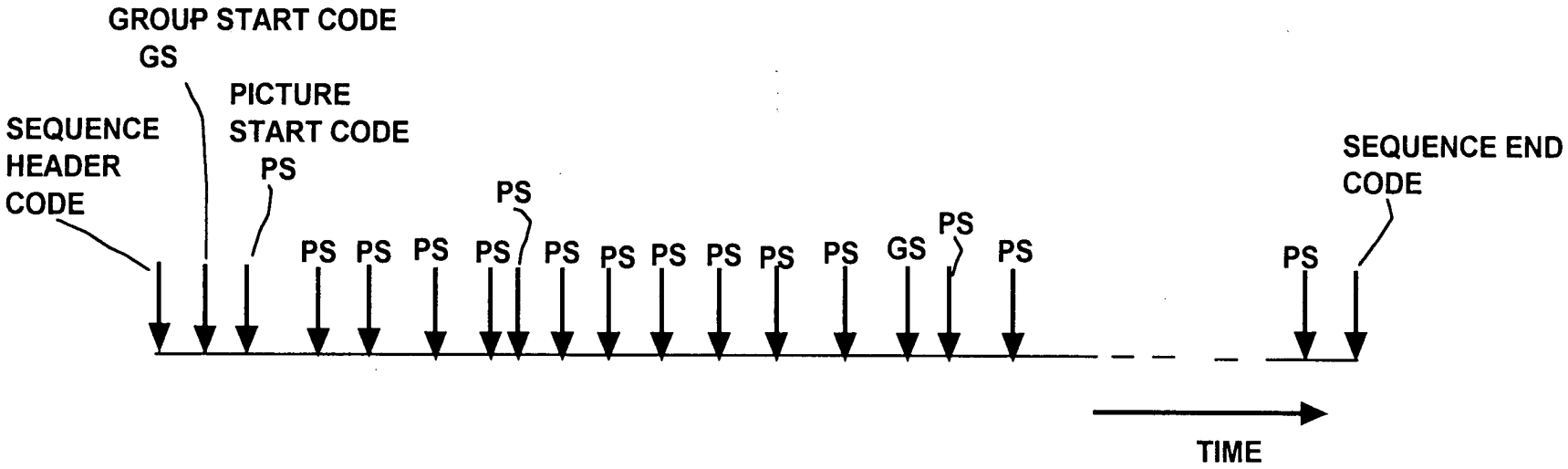
GROUP START CODE
GS
PICTURE
START CODE
PS
SEQUENCE
HEADER
CODE
SEQUENCE END
CODE
PS
GS
TIME

FIG. 10A

PICTURE LAYER

SYNTAX	NUMBER OF BITS
Picture () {	
picture_start_code	32
Temporal reference	10
picture_coding _type	3
vbv_delay	16
if (picture_coding_type==2) ||	
(picture_coding_type==3)) {	
full_pel_forward_vector	1
forward_f_code	3
}	
if (picture_coding_type==3) {	
full_pel_backward_vector	
backward_f_code	
}	
while (nextbits () ='1') {	
extra_bit_picture	1
extra_information_picture	8
}	
extra_bit_picture	1
next_start_code ()	
if (nextbits () ==extension_start_code) {	
extension_start_code	
while (nextbits () ! ='0000 0000 0000 0000 0000 0001'){	32
picture_extension_data	8
}	
Next_start_code ()	
}	
if (nextbits () ==user_data_start_code) {	
user_data_start_code	32
while (nextbits () ! ='0000 0000 0000 0000 0000 0001'){	
user_data	8
}	
Next_start_code ()	
}	
do {	
Slice ()	
} while (nextbits () ==slice_start_code	

FIG. 10B

SLICE LAYER

SYNTAX	NUMBER OF BITS
slice () {	
slice_start_code	32
quantizer_scale	5
while (nextbits () =='1') {	
extra_bit_slice	1
extra_information_slice	8
}	
extra_bit_slice	1
do {	
Macroblock ()	
} while (nextbits () != '000 0000 0000 0000 0000 0000')	
next_start_code()	
}	

FIG. 10C

MACROBLOCK LAYER

SYNTAX	NUMBER OF BITS
macroblock () {	
while (nextbits () ='0000 0001 111')	
macroblock stuffing	11
while (nextbits () ='0000 0001 000')	
macroblock escape	11
macroblock_address_increment	1-11
macroblock_type	1-6
if (macroblock_quant)	
Quantizer_scale	5
if (macroblock_motion_forward) {	
motion_horizontal_forward_code	1-11
if ((forward_f !=1) &&	
(motion_horizontal_forward_code!=0))	
motion_horizontal_forward_r	1-6
motion_vertical_forward_code	1-11
if ((forward_f !=1) &&	
(motion_vertical_forward_code!=0))	
motion_vertical_forward_r	1-6
}	
if ((macroblock_motion_backward) {	
motion_horizontal_backward_code	1-11
if (backward_f !=1)&&	
(motion_horizontal_backward_code !=0)	
motion_horizontal_backward_r	1-6
motion_vertical_backward_code	1-11
if (backward_f !=1)&&	
(motion_vertical_backward_code !=0)	
motion_vertical_backward_r	1-6
}	
if (macroblock_pattern)	
coded_block_pattern	3-9
for (i=0; i<6; i++)	
Block(i)	
if (picture_coding_type ==4)	
End_of_macroblock	1

FIG. 10D

BLOCK LAYER

SYNTAX	NUMBER OF BITS
block (i) {	
if (pattern_code[i]) {	
if (macroblock_intra) {	
if (i<4) {	
dct_size_luminance	2-7
if (dct_size_luminance !=0)	
dct_dc_differential	1-8
}	
else {	
dct_size_chrominance	2-8
if (dct_size_chrominance !=0)	
dct_dc_differential	1-8
}	
}	
else {	
dct_coeff_first	2-28
}	
if (picture_coding_type !=4) {	
while (nextbits() != '10')	
dct_coeff_next	3-28
end_of_block	2
}	
}	
}	

FIG. 11

CODE	SYNTAX	NUMBER OF BITS
0000 0000 0000 0000 0000 0001 0000 0000	picture_start_code	32
(From MPEG data stream)	temporal reference	10
010 (for P-picture)	picture_coding _type	3
(From MPEG data stream)	vbv_delay	16
0	full_pel_forward_code	1
001	forward_f_code	3
0000 000	stuffing	7
0000 0000 0000 0000 0000 0001 0000 0001	slice_start_code	32
0000 1	quantizer scale	5
1	macroblock_address _increment	1
001	macroblock_type	3
0	motion_horizontal_forward_code	1
0	motion_horizontal_backward_code	1
0000 0001 000 (x11)	macroblock_escape(x11)	121
0000 0011 001	macroblock_address_increment	11
001	macroblock_type	3
0	motion_horizontal_forward_code	1
0	motion_horizontal_backward_code	1
0000	stuffing	4
	TOTAL	256 bits

FIG. 12

CODE	SYNTAX	NUMBER OF BITS
0000 0000 0000 0000 0000 0001 0000 0000	picture_start_code	32
(From MPEG data stream)	temporal reference	10
011 (for B-picture)	picture_coding _type	3
(From MPEG data stream)	vbv_delay	16
0	full_pel_forward_code	1
001	forward_f_code	3
0000 000	stuffing	7
0000 0000 0000 0000 0000 0001 0000 0001	slice_start_code	32
0000 1	quantizer scale	5
1	macroblock_address _increment	1
001	macroblock_type	3
0	motion_horizontal_forward_code	1
0	motion_horizontal_backward_code	1
0000 0001 000 (x11)	macroblock_escape(x11)	121
0000 0011 001	macroblock_address_increment	11
001	macroblock_type	3
0	motion_horizontal_forward_code	1
0	motion_horizontal_backward_code	1
0000	stuffing	4
	TOTAL	256 bits

FIG.13A

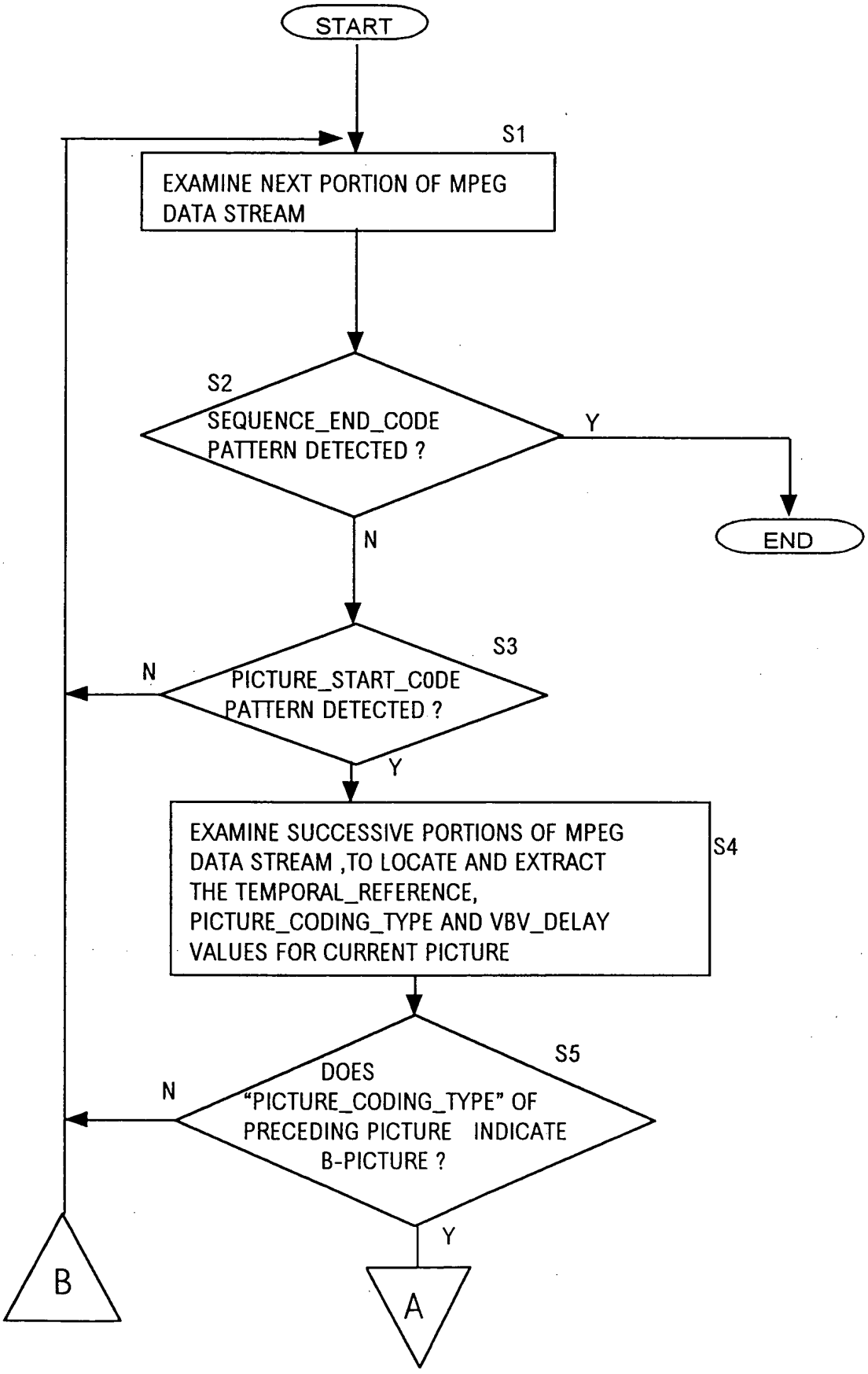
START
S1
EXAMINE NEXT PORTION OF MPEG DATA STREAM
S2
SEQUENCE_END_CODE PATTERN DETECTED ?
Y
END
N
S3
N
PICTURE_START_CODE PATTERN DETECTED ?
Y
EXAMINE SUCCESSIVE PORTIONS OF MPEG DATA STREAM ,TO LOCATE AND EXTRACT THE TEMPORAL_REFERENCE, PICTURE_CODING_TYPE AND VBV_DELAY VALUES FOR CURRENT PICTURE
S4
S5
DOES "PICTURE_CODING_TYPE" OF PRECEDING PICTURE INDICATE B-PICTURE ?
N
Y
B
A

FIG.13B

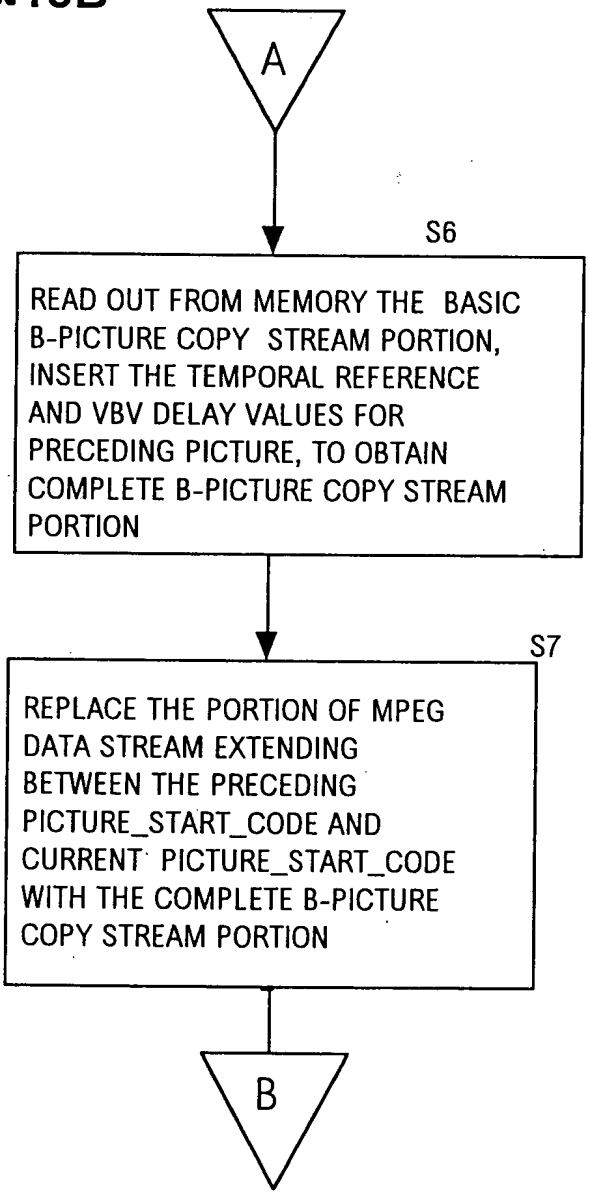
A
S6
READ OUT FROM MEMORY THE BASIC B-PICTURE COPY STREAM PORTION, INSERT THE TEMPORAL REFERENCE AND VBV DELAY VALUES FOR PRECEDING PICTURE, TO OBTAIN COMPLETE B-PICTURE COPY STREAM PORTION
S7
REPLACE THE PORTION OF MPEG DATA STREAM EXTENDING BETWEEN THE PRECEDING PICTURE_START_CODE AND CURRENT PICTURE_START_CODE WITH THE COMPLETE B-PICTURE COPY STREAM PORTION
B

FIG. 14A

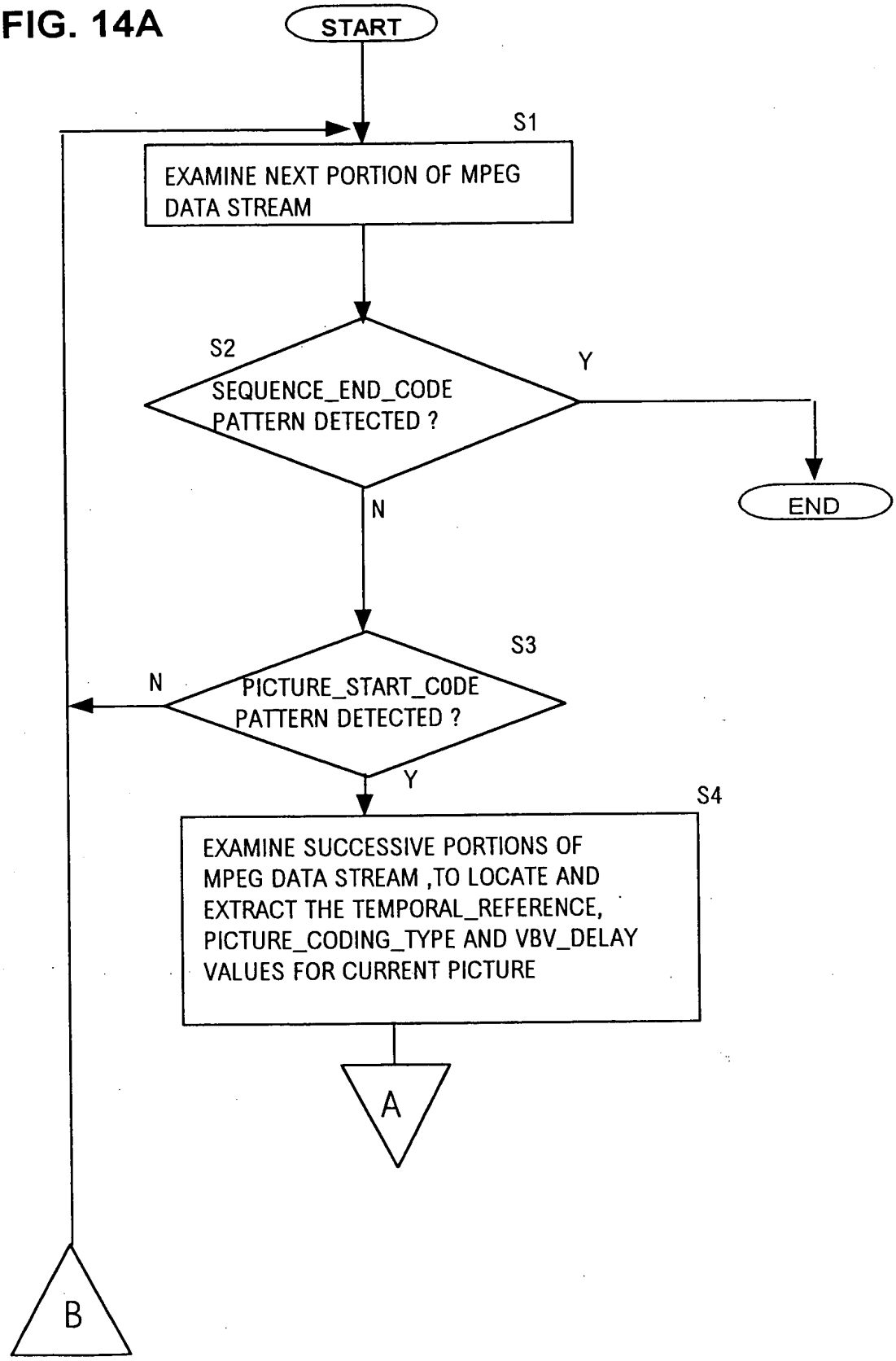
START
S1
EXAMINE NEXT PORTION OF MPEG DATA STREAM
S2
SEQUENCE_END_CODE PATTERN DETECTED ?
Y
END
N
S3
N
PICTURE_START_CODE PATTERN DETECTED ?
Y
S4
EXAMINE SUCCESSIVE PORTIONS OF MPEG DATA STREAM ,TO LOCATE AND EXTRACT THE TEMPORAL_REFERENCE, PICTURE_CODING_TYPE AND VBV_DELAY VALUES FOR CURRENT PICTURE
A
B

FIG.14B

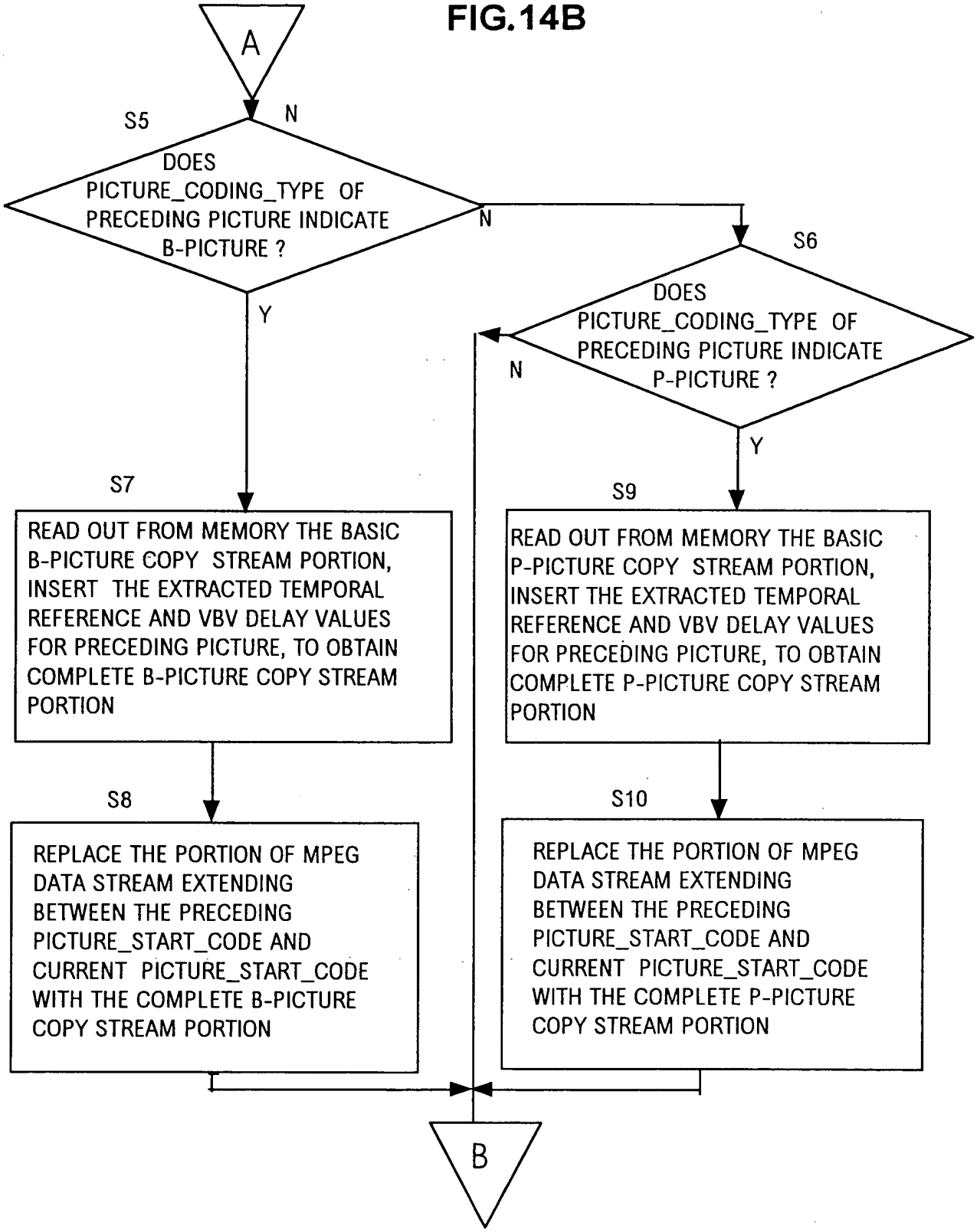
A
S5
N
DOES PICTURE_CODING_TYPE OF PRECEDING PICTURE INDICATE B-PICTURE ?
N
S6
DOES PICTURE_CODING_TYPE OF PRECEDING PICTURE INDICATE P-PICTURE ?
N
Y
Y
S7
READ OUT FROM MEMORY THE BASIC B-PICTURE COPY STREAM PORTION, INSERT THE EXTRACTED TEMPORAL REFERENCE AND VBV DELAY VALUES FOR PRECEDING PICTURE, TO OBTAIN COMPLETE B-PICTURE COPY STREAM PORTION
S8
REPLACE THE PORTION OF MPEG DATA STREAM EXTENDING BETWEEN THE PRECEDING PICTURE_START_CODE AND CURRENT PICTURE_START_CODE WITH THE COMPLETE B-PICTURE COPY STREAM PORTION
S9
READ OUT FROM MEMORY THE BASIC P-PICTURE COPY STREAM PORTION, INSERT THE EXTRACTED TEMPORAL REFERENCE AND VBV DELAY VALUES FOR PRECEDING PICTURE, TO OBTAIN COMPLETE P-PICTURE COPY STREAM PORTION
S10
REPLACE THE PORTION OF MPEG DATA STREAM EXTENDING BETWEEN THE PRECEDING PICTURE_START_CODE AND CURRENT PICTURE_START_CODE WITH THE COMPLETE P-PICTURE COPY STREAM PORTION
B

FIG. 15

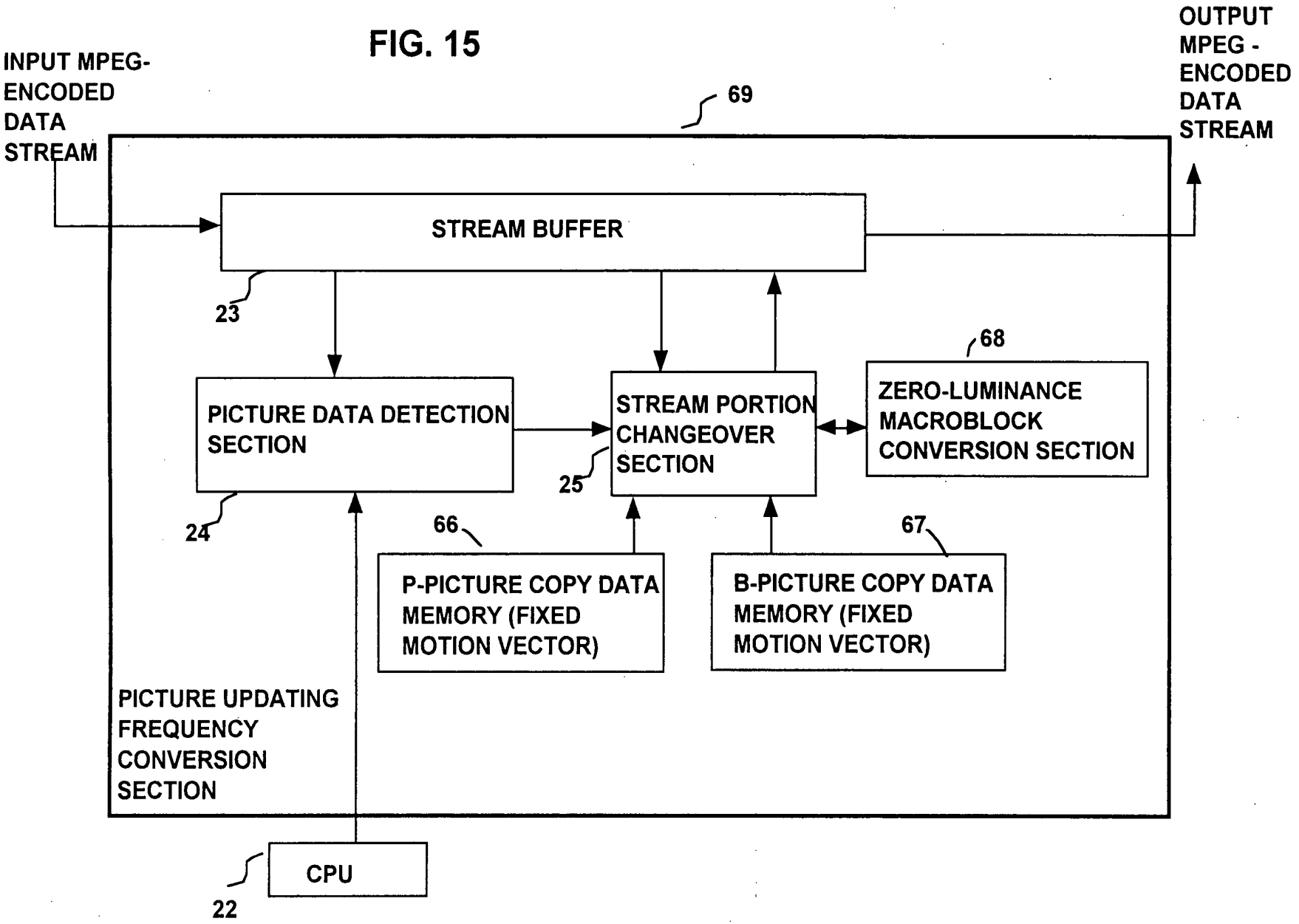
INPUT MPEG-ENCODED DATA STREAM
OUTPUT MPEG-ENCODED DATA STREAM
69
STREAM BUFFER
23
PICTURE DATA DETECTION SECTION
24
STREAM PORTION CHANGEOVER SECTION
25
68
ZERO-LUMINANCE MACROBLOCK CONVERSION SECTION
66
P-PICTURE COPY DATA MEMORY (FIXED MOTION VECTOR)
67
B-PICTURE COPY DATA MEMORY (FIXED MOTION VECTOR)
PICTURE UPDATING FREQUENCY CONVERSION SECTION
CPU
22

FIG. 16A

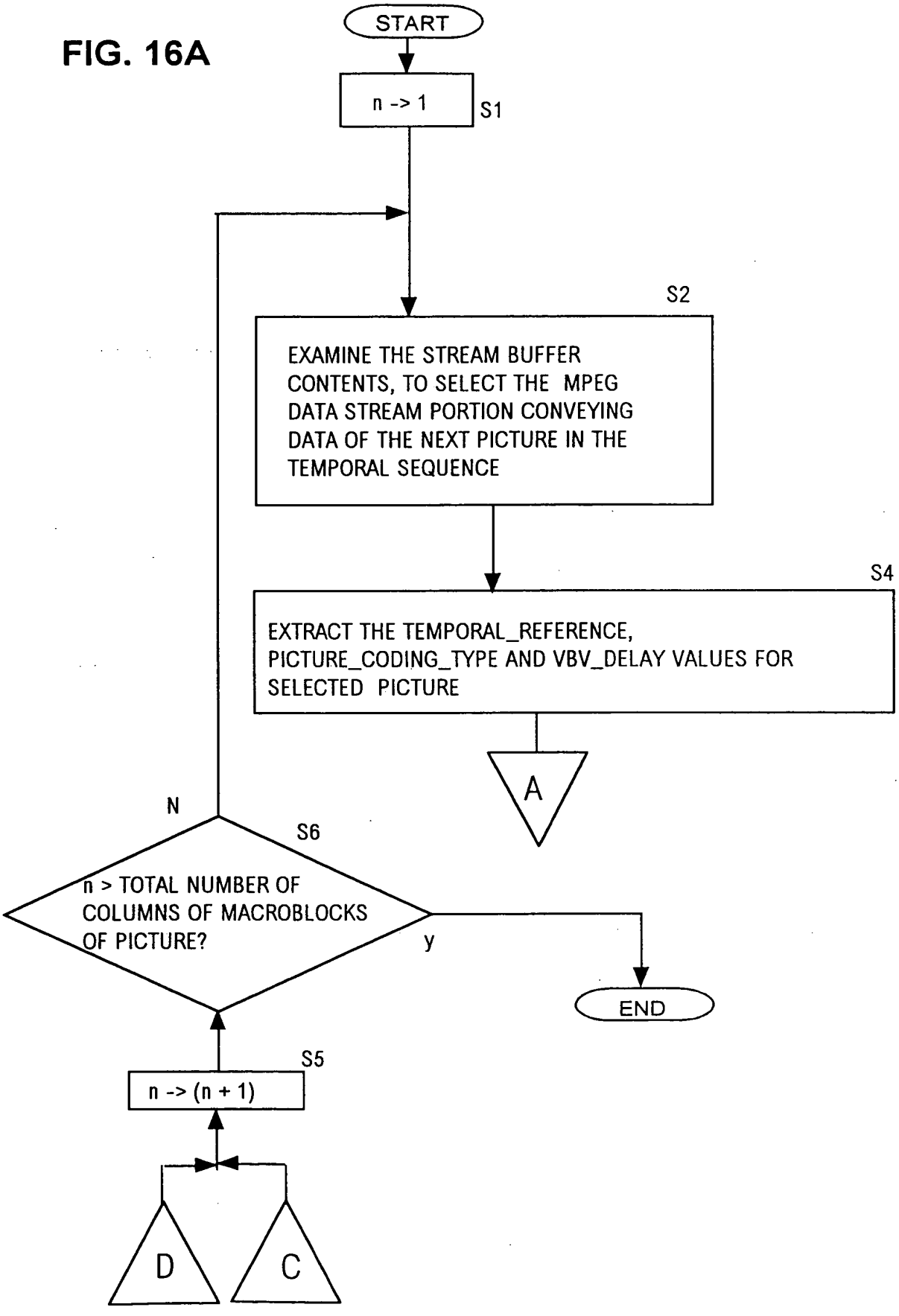
START
n -> 1
S1
S2
EXAMINE THE STREAM BUFFER CONTENTS, TO SELECT THE MPEG DATA STREAM PORTION CONVEYING DATA OF THE NEXT PICTURE IN THE TEMPORAL SEQUENCE
S4
EXTRACT THE TEMPORAL_REFERENCE, PICTURE_CODING_TYPE AND VBV_DELAY VALUES FOR SELECTED PICTURE
A
N
S6
n > TOTAL NUMBER OF COLUMNS OF MACROBLOCKS OF PICTURE?
y
END
S5
n -> (n + 1)
D
C

FIG. 16B

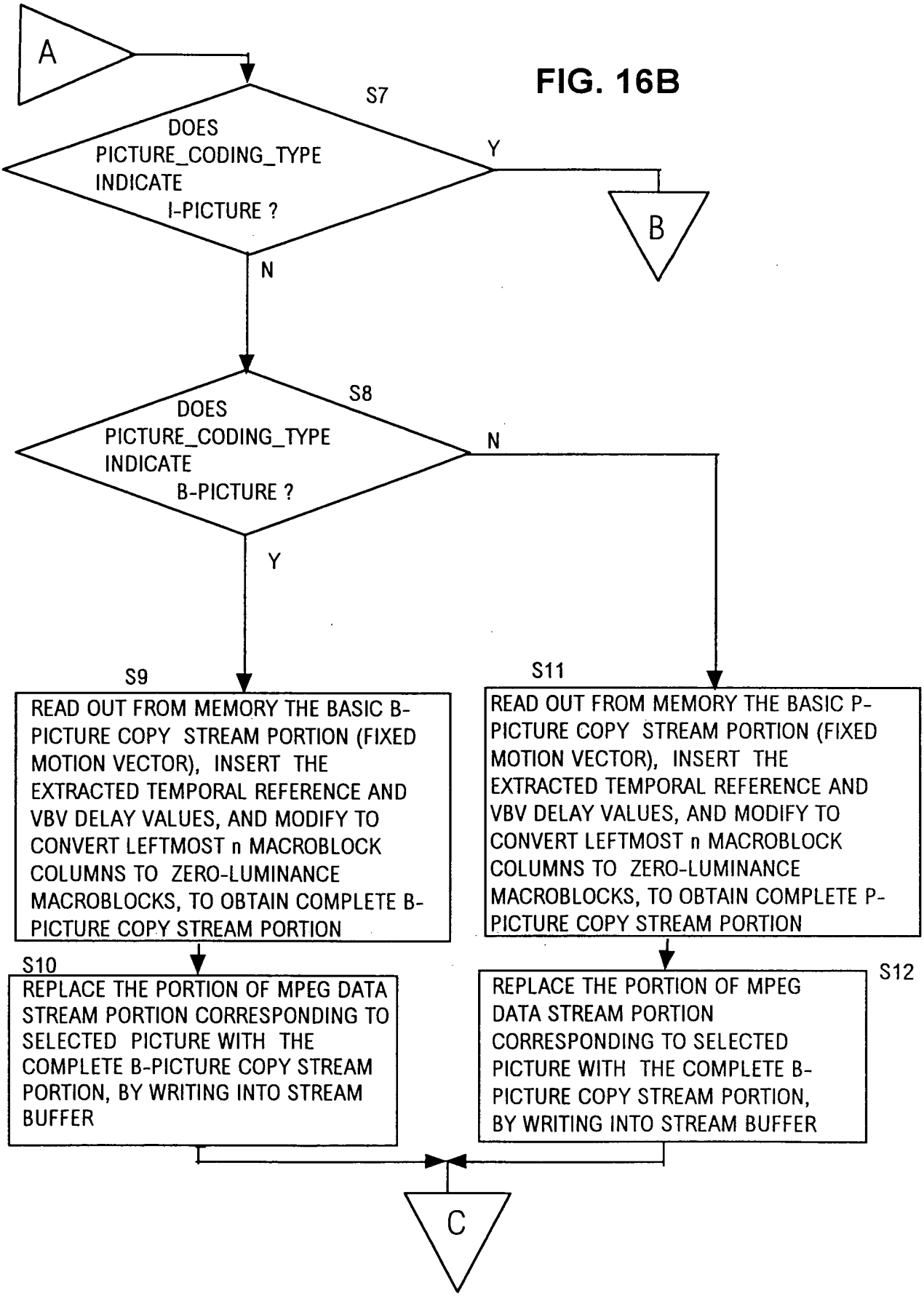
A
S7
DOES PICTURE_CODING_TYPE INDICATE I-PICTURE ?
Y
B
N
S8
DOES PICTURE_CODING_TYPE INDICATE B-PICTURE ?
N
Y
S9
READ OUT FROM MEMORY THE BASIC B-PICTURE COPY STREAM PORTION (FIXED MOTION VECTOR), INSERT THE EXTRACTED TEMPORAL REFERENCE AND VBV DELAY VALUES, AND MODIFY TO CONVERT LEFTMOST n MACROBLOCK COLUMNS TO ZERO-LUMINANCE MACROBLOCKS, TO OBTAIN COMPLETE B-PICTURE COPY STREAM PORTION
S10
REPLACE THE PORTION OF MPEG DATA STREAM PORTION CORRESPONDING TO SELECTED PICTURE WITH THE COMPLETE B-PICTURE COPY STREAM PORTION, BY WRITING INTO STREAM BUFFER
S11
READ OUT FROM MEMORY THE BASIC P-PICTURE COPY STREAM PORTION (FIXED MOTION VECTOR), INSERT THE EXTRACTED TEMPORAL REFERENCE AND VBV DELAY VALUES, AND MODIFY TO CONVERT LEFTMOST n MACROBLOCK COLUMNS TO ZERO-LUMINANCE MACROBLOCKS, TO OBTAIN COMPLETE P-PICTURE COPY STREAM PORTION
S12
REPLACE THE PORTION OF MPEG DATA STREAM PORTION CORRESPONDING TO SELECTED PICTURE WITH THE COMPLETE B-PICTURE COPY STREAM PORTION, BY WRITING INTO STREAM BUFFER
C

FIG. 16C

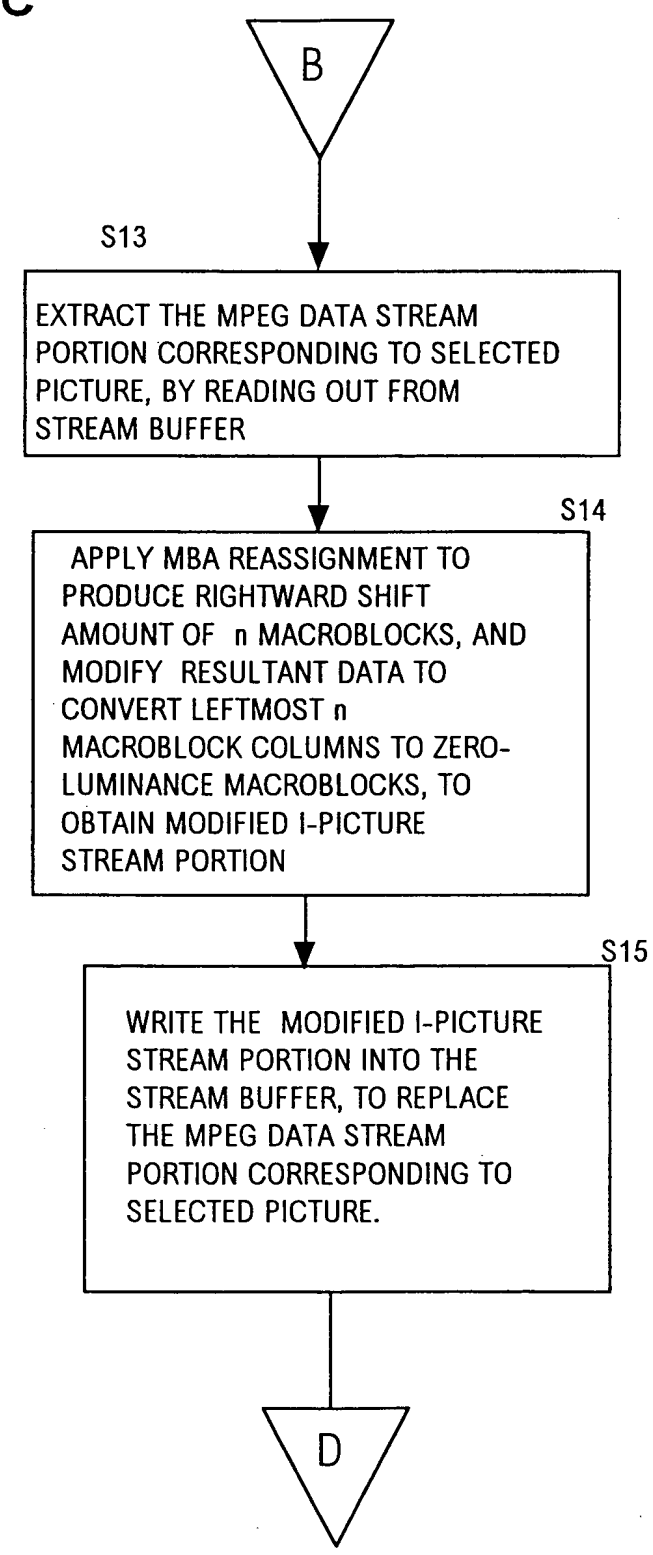
B
S13
EXTRACT THE MPEG DATA STREAM PORTION CORRESPONDING TO SELECTED PICTURE, BY READING OUT FROM STREAM BUFFER
S14
APPLY MBA REASSIGNMENT TO PRODUCE RIGHTWARD SHIFT AMOUNT OF n MACROBLOCKS, AND MODIFY RESULTANT DATA TO CONVERT LEFTMOST n MACROBLOCK COLUMNS TO ZERO-LUMINANCE MACROBLOCKS, TO OBTAIN MODIFIED I-PICTURE STREAM PORTION
S15
WRITE THE MODIFIED I-PICTURE STREAM PORTION INTO THE STREAM BUFFER, TO REPLACE THE MPEG DATA STREAM PORTION CORRESPONDING TO SELECTED PICTURE.
D

FIG. 17

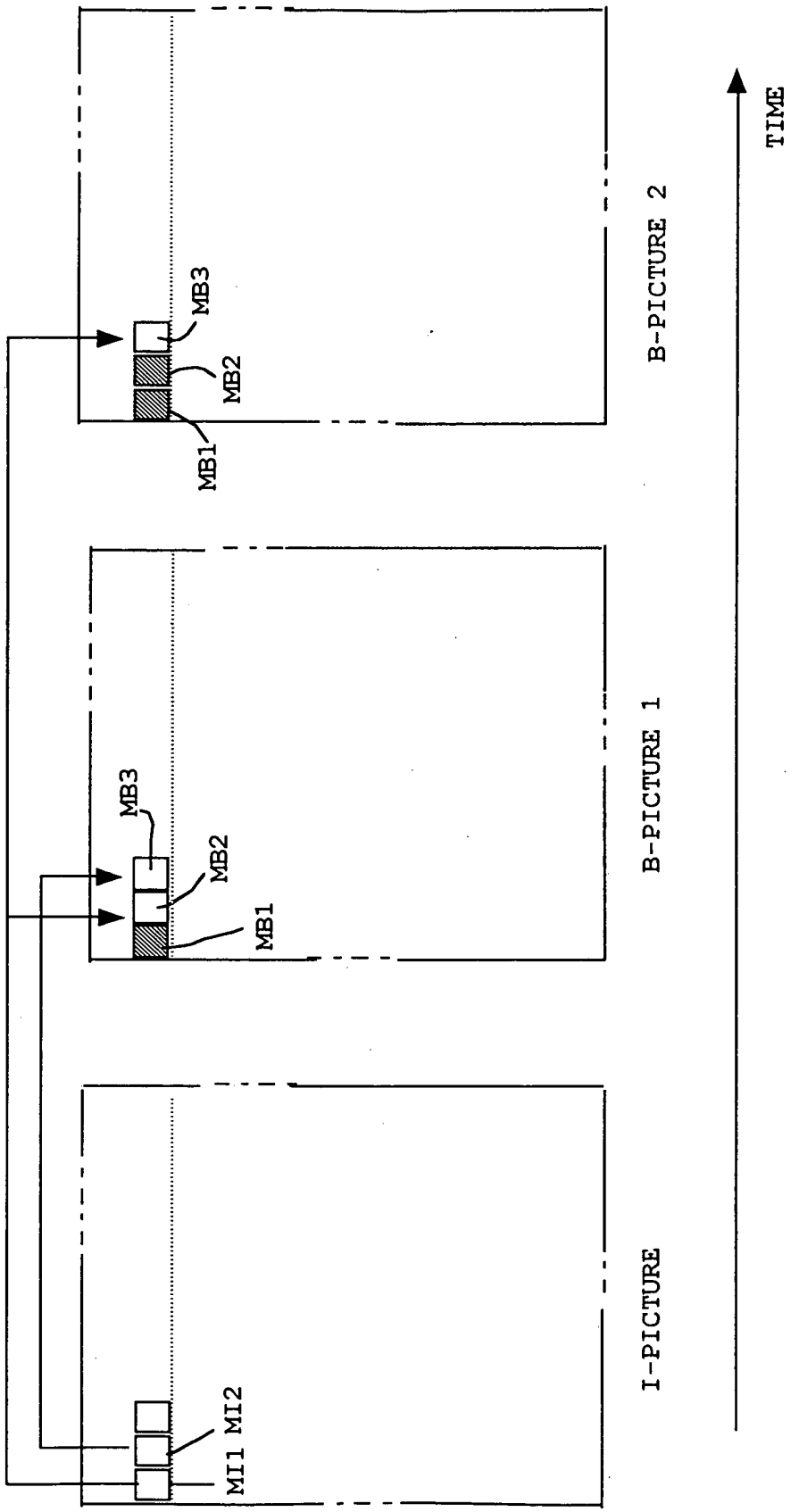
MB3
MB2
MB1
B-PICTURE 2
MB3
MB2
MB1
B-PICTURE 1
MI1 MI2
I-PICTURE
TIME

FIG. 18

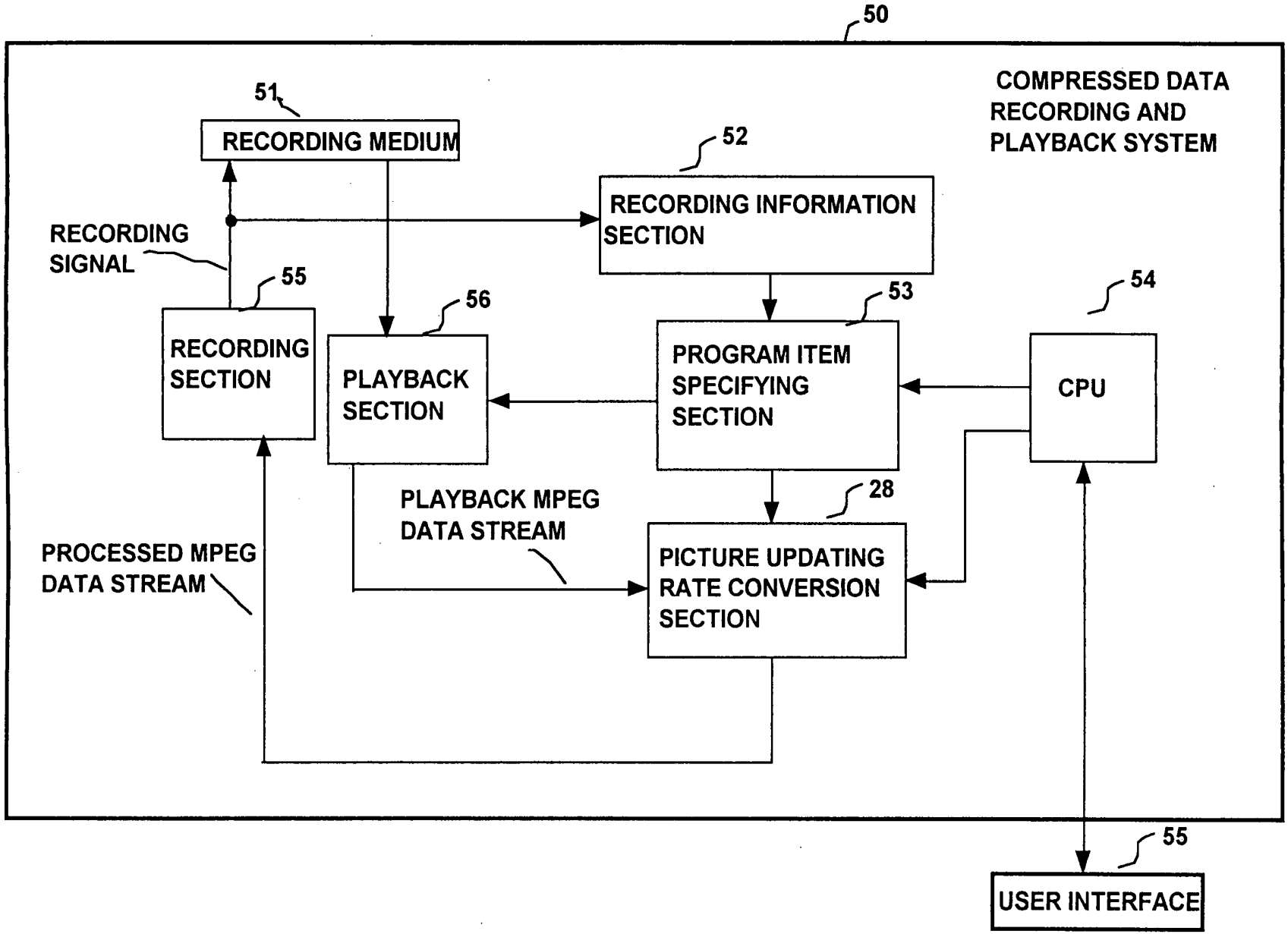
50
COMPRESSED DATA RECORDING AND PLAYBACK SYSTEM
51
RECORDING MEDIUM
52
RECORDING INFORMATION SECTION
RECORDING SIGNAL
55
RECORDING SECTION
56
PLAYBACK SECTION
53
PROGRAM ITEM SPECIFYING SECTION
54
CPU
PLAYBACK MPEG DATA STREAM
28
PROCESSED MPEG DATA STREAM
PICTURE UPDATING RATE CONVERSION SECTION
55
USER INTERFACE

FIG. 19 PRIOR ART

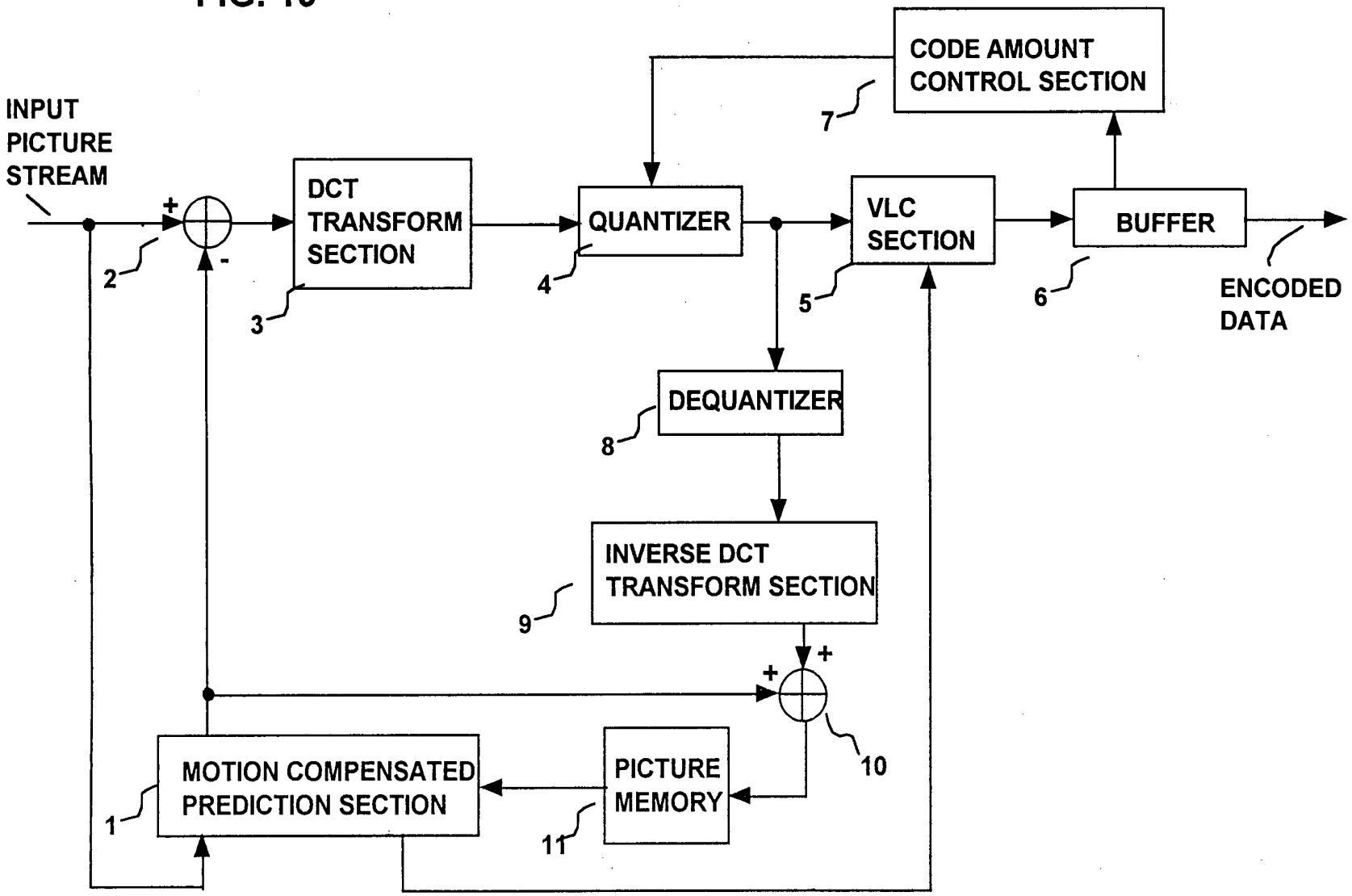

FIG. 20 PRIOR ART